Session 15
Software bug report

xfzhang@suda.edu.cn

Review

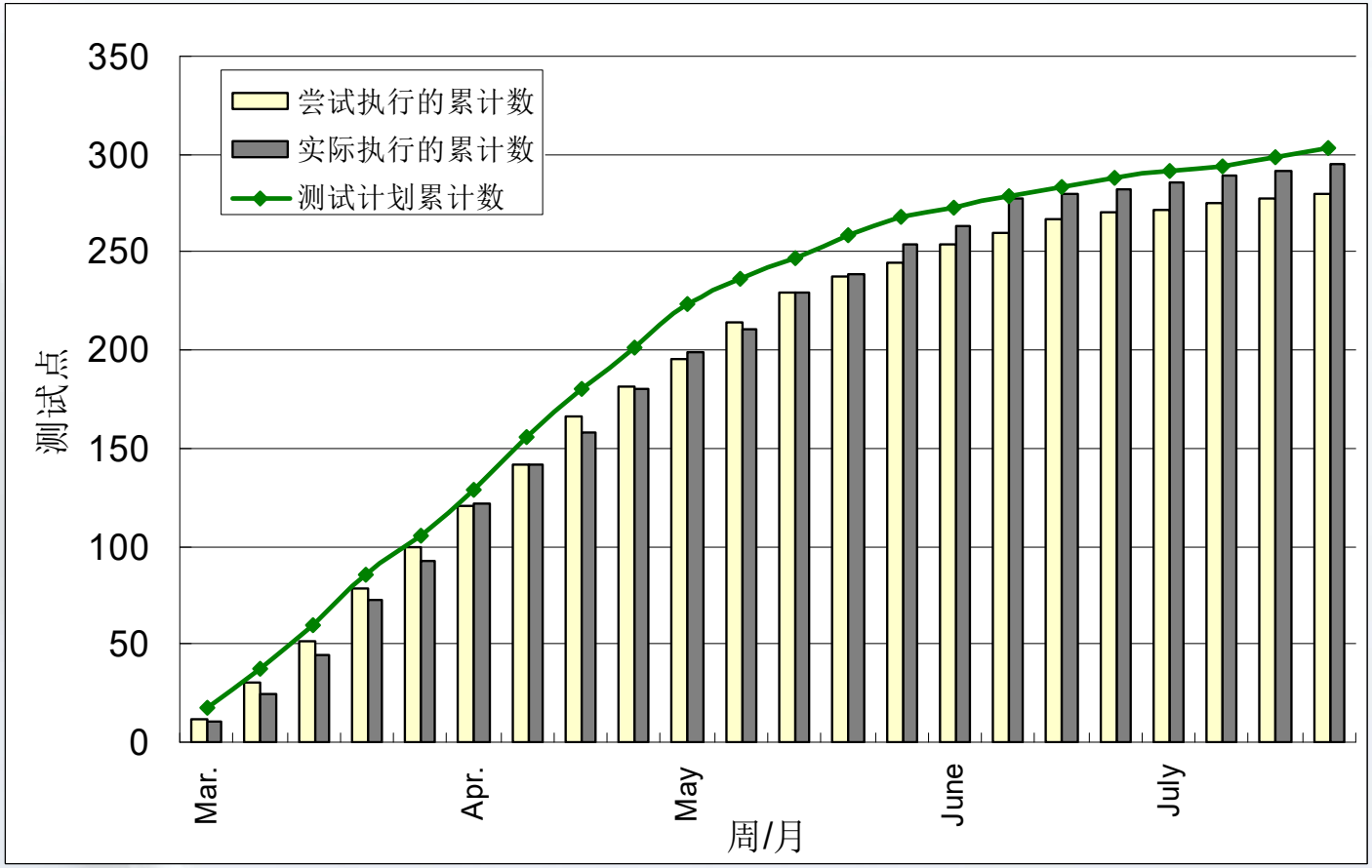
- Introduction & Preliminary
- Testing Methods
  - White box
    - Logic Coverage/Basis Path/Loop/Data flow
  - Black box
    - BVA / EP / DT / CEG / CT
- Testing Process
  - U-I-S-A
  - Regression Testing

软件缺陷报告

- 软件测试进度
- 软件缺陷的描述
  - 软件缺陷的生命周期
  - 严重性和优先级
  - 缺陷的其它属性
  - 完整的缺陷信息
  - 缺陷描述的基本要求
  - 缺陷报告的示例
- 软件缺陷跟踪和分析

测试进度的 S曲线法

进度S曲线法通过对计划中、尝试的与实际的进度三者对比来实现的，其采用的基本数据主要是测试用例或测试点的数量



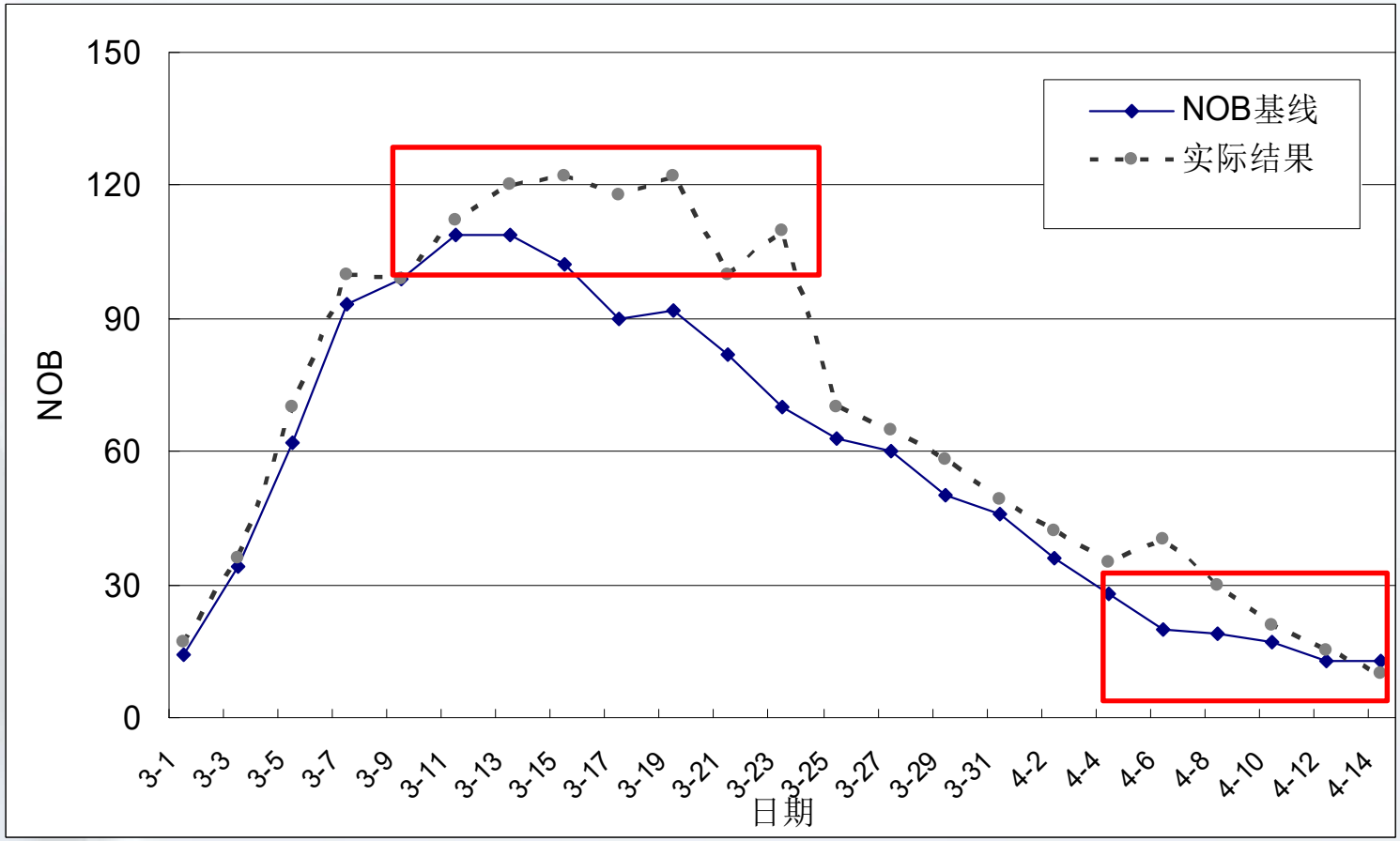
初始阶段

紧张阶段

成熟阶段

测试进度的NOB曲线法

NOB，Number of Open Bug



软件缺陷报告

- 软件测试进度
- 软件缺陷的描述
  - 软件缺陷的生命周期
  - 严重性和优先级
  - 缺陷的其它属性
  - 完整的缺陷信息
  - 缺陷描述的基本要求
  - 缺陷报告的示例
- 软件缺陷跟踪和分析

对缺陷会关心哪些问题？

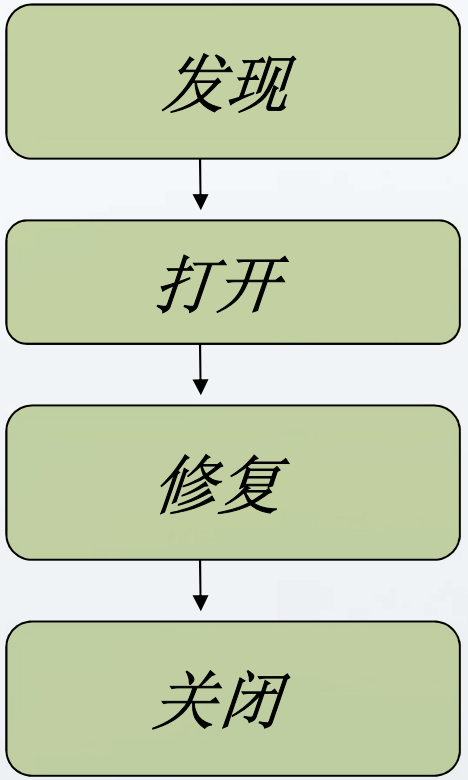
- 描述？
- 是否严重？
- 是否需要修正？
- 当前状态？
- ？

软件缺陷的生命周期

- 软件缺陷生命周期指的是一个软件缺陷被发现、报告到这个缺陷被修复、验证直至最后关闭的完整过程
- 缺陷生命周期是各类开发人员一起参与、协同测试的过程。
- 软件缺陷一旦发现，便进入严密监控之中，直至软件缺陷生命周期终结，这样即可保证在较短的时间内高效率地关闭所有的缺陷，缩短软件测试的进程，提高软件质量，同时减少开发、测试和维护成本。

基本的缺陷生命周期

- 发现-打开：测试人员找到软件缺陷并将软件缺陷提交给开发人员。
- 打开-修复：开发人员再现、修复缺陷，然后提交给测试人员去验证。
- 修复-关闭：测试人员验证修复过的软件，关闭已不存在的缺陷。



复杂的缺陷生命周期

发现软件缺陷

打开

测试员找到并登记软件缺陷

软件缺陷被移交到程序员

打开

程序员认为软件缺陷微不足道

软件缺陷移交到项目管理员

以不修复形式解决

项目管理员认为软件缺陷不重要

软件缺陷移交到测试员

打开

测试员不同意，找出通用失败案例

软件缺陷移交到项目管理员

打开

项目管理员现在同意软件缺陷需要修复

软件缺陷移交到程序员

以修复形式解决

程序员修复软件缺陷

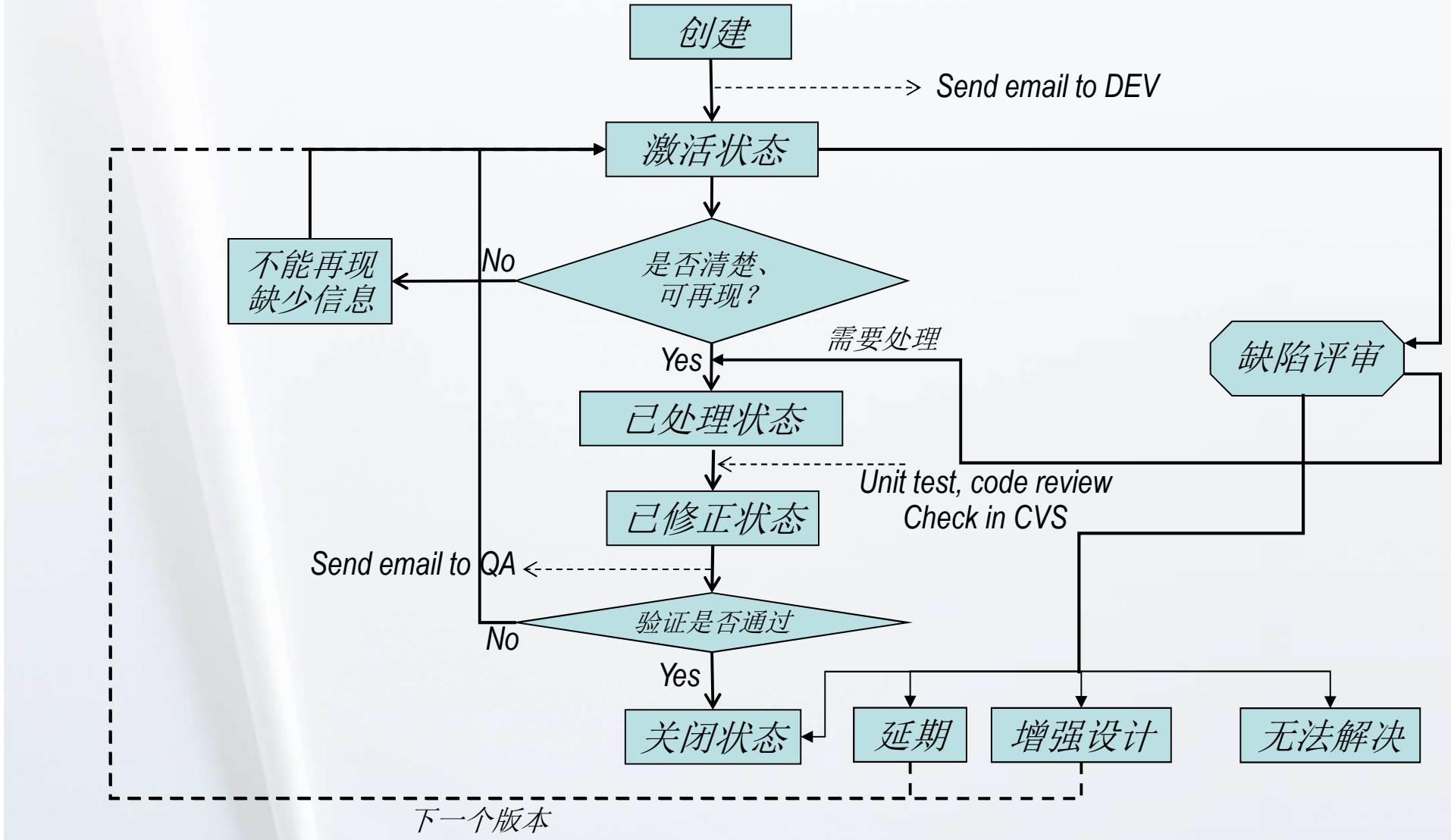
软件缺陷移交到测试员

关闭

测试员确认软件缺陷得以修复

测试员关闭软件缺陷

实际的缺陷生命周期


创建
Send email to DEV
激活状态
不能再现
缺少信息
No
是否清楚、
可再现？
需要处理
缺陷评审
Yes
已处理状态
Unit test, code review
Check in CVS
已修正状态
Send email to QA
验证是否通过
No
Yes
关闭状态
延期
增强设计
无法解决
下一个版本

实例

start

1. Open a bug
2. Dev checks mail & Review bug
3. Duplicate the bug
4. Debug
5. Check out code
6. Fix bug
7. Code Review
8. Unit test
9. Check in code
10. Build a package
11. Upload package
12. Installation/configuration
13. Verify fixed bugs
14. Change bug status to close

end

14 Steps

软件缺陷报告

- 软件测试进度
- 软件缺陷的描述
  - 软件缺陷的生命周期
  - 严重性和优先级
  - 缺陷的其它属性
  - 完整的缺陷信息
  - 缺陷描述的基本要求
  - 缺陷报告的示例
- 软件缺陷跟踪和分析

严重性和优先级

- 严重性（severity）衡量缺陷对客户满意度的影响程度

  致命的（fatal）、严重的（critical）、一般的（major）、微小的（minor）

严重性级别：定义是相对的

① 致命错误：导致系统崩溃、数据丢失、数据毁坏等；

②严重错误：功能或特性没有实现，主要功能部分丧失，次要功能完全丧失等；

③一般性错误：操作性错误、错误结果、遗漏功能等；

④次要错误：错别字、用户接口布局、罕见故障等。

严重性和优先级

- 优先级(Priority)：指缺陷被修复的紧急程度。

缺陷优先级	描述
立即解决(P1级)	缺陷导致系统几乎不能使用或测试不能继续，需立即修复
高优先级(P2级）	缺陷严重，影响测试，需要优先考虑
正常排队(P3级）	缺陷需要正常排队等待修复
低优先级(P4级）	缺陷可以在开发人员有时间的时候被纠正。

严重性和优先级

启动系统就崩溃：严重性① ，优先级① ；

界面中按钮下移：严重性③，优先级④ 。

极少发生的数据毁坏：严重性①，优先级③；

安装指南中的错别字：严重性③，优先级②；

项目不同，严重性和优先级信息也不同。

软件缺陷的优先级在项目期间会发生变化。

缺陷的其它属性

- 缺陷**标识**（ID）
- 缺陷**类型**（type），如功能、UI、性能、文档
- 缺陷产生**可能性**（frequency）/可再现的概率
- 缺陷**来源**（source）：需求、设计、编码
- 缺陷**原因**（cause）：数据格式、计算错误、接口参数、变量定义与引用等
- ……

见 P.297 诸表

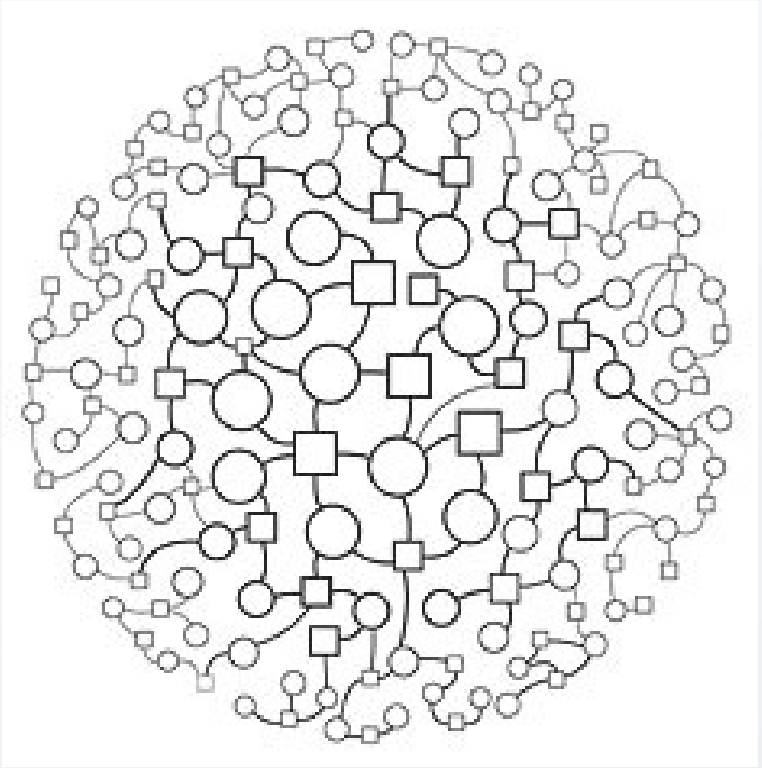
基本的缺陷信息

- 步骤：提供了如何重复当前缺陷的准确描述，应简明而完备、清楚而准确。这些信息对开发人员是关键的，视为修复缺陷的向导
- 期望结果：与测试用例标准或设计规格说明书或用户需求等一致，达到软件预期的功能。是验证缺陷的依据。
- 实际结果：实际执行测试的结果，不同于期望结果，从而确认缺陷的存在

还需要什么重要的信息？

其它信息

- 产品信息
- 版本信息
- 图片
- Trace Log
- 录制这个操作过程
- …



完整的缺陷信息

- ID
- 标题
- 前提
- 环境
- 操作步骤
- 期望结果
- 实际结果
- 频率
- 严重程度
- 优先级
- 类型
- 缺陷提交人
- 缺陷指定解决人
- 来源
- 产生原因
- 产品信息
- 状态
- 提交时间
- 修正时间
- 验证时间
- 所属项目/模块
- ……

软件缺陷报告

任何一个缺陷跟踪系统的核心都是“软件缺陷报告”，一份软件缺陷报告详细信息如表：

软件缺陷项目列表

分类	项目	描述
可跟踪信息	缺陷ID	唯一的、自动产生的缺陷ID，用于识别、跟踪、查询
软件缺陷基本信息	缺陷状态	可分为“打开或激活的”、“已修正”、“关闭”等
	缺陷标题	描述缺陷的最主要信息
	缺陷的严重程度	一般分为“致命”、“严重”、“一般”、“较小”等四种程度
	缺陷的优先级	描述处理缺陷的紧急程度， 1是优先级最高的等级，2是正常的，3是优先级最低的
	缺陷的产生频率	描述缺陷发生的可能性1%-100%
	缺陷提交人	缺陷提交人的名字（会和邮件地址联系起来），一般就是发现缺陷的测试人员或其他人员
	缺陷提交时间	缺陷提交的时间

软件缺陷报告

软件缺陷基本信息	缺陷所属项目/模块	缺陷所属的项目和模块，最好能较精确的定位至模块
	缺陷指定解决人	估计修复这个缺陷的开发人员，在缺陷状态下由开发组长指定相关的开发人员；也会自动和该开发人员的邮件地址联系起来，并自动发出邮件
	缺陷指定解决时间	开发管理员指定的开发人员修改此缺陷的时间
	缺陷验证人	验证缺陷是否真正被修复的测试人员；也会和邮件地址联系起来
	缺陷验证结果描述	对验证结果的描述（通过、不通过）
	缺陷验证时间	对缺陷验证的时间
缺陷的详细描述	步骤	对缺陷的操作过程，按照步骤，一步一步地描述
	期望的结果	按照设计规格说明书或用户需求，在上述步骤之后，所期望的结果，即正确的结果
	实际发生的结果	程序或系统实际发生的结果，即错误的结果
测试环境说明	测试环境	对测试环境描述，包括操作系统、浏览器、网络带宽、通讯协议等
必要的附件	图片、Log文件	对于某些文字很难表达清楚的缺陷，使用图片等附件是必要的；对于软件崩溃现象，需要捕捉日志文件作为附件提供给开发人员。

IEEE829－1998 软件测试文档编制标准

软件缺陷报告模板

目录

IEEE软件缺陷报告模板

软件缺陷报告文档

公司名称：_____ BUG 报告：_____ BUG# ：_____

软件：_____ 版本：_____

测试员：_____日期：_____

严重性：_____优先级：_____ 是否会重现：是 否

标题：_____

描述：_____

_____

_____

解决方法：_____

解决日期：_____ 解决人：_____ 版本号：_____

解决描述：_____

_____

_____

重新测试人：_____测试版本号：_____测试日期：_____

重新测试描述：_____

_____

_____

签名：

策划：_____测式：_____

编程：_____项目管理：_____

销售：_____技术支持：_____

缺陷描述的基本要求

- 单一准确
- 可以再现
- 完整统一
- 短小简练
- 特定条件
- 补充完善
- 不做评价

示例

优秀的缺陷报告

(1)重现步骤 ：

a) 打开一个编辑文字的软件并且创建一个新的文档（这个文件可以录入文字）

b) 在这个文件里随意录入一两行文字

c) 选中一两行文字，通过选择Font 菜单然后选择Arial字体格式

d) 一两行文字变成了无意义的乱字符

(2)期望结果：当用户选择已录入的文字并改变文字格式的时候，文本应该显示正确的文字格式不会出现乱字符显示。

(3)实际结果:它是字体格式的问题，如果改变文字格式成Arial之前，你保存文件，缺陷不会出现。缺陷仅仅发生在Windows98并且改变文字格式成其它的字体格式，文字是显示正常的。

见所附的图片<有一个链接，点击即可看到>

散漫的缺陷报告的示例

重现步骤：

- 在WindowXP上打开一个编辑文字的软件并且编辑存在文件
- 文件字体显示正常
- 我添加了图片，这些图片显示正常
- 在此之后，我创建了一个新的文档
- 在这个文档中我随意录入了大量的文字
- 在我录入这些文字之后，选择几行文字.并且通过选择Font 菜单然后选择Arial字体格式改变文字的字体。
- 有三次我重现了这个缺陷 ？？？
- 我在Solaris操作系统运行这些步骤，没有任何问题。
- 我在Mac操作系统运行这些步骤，没有任何问题。

期望结果：当用户选择已录入的文字并改变文字格式的时候，文本应该显示正确的文字格式不会出现乱字符显示。

实际结果：我试着选择少量的不同的字体格式，但是只有Arial字体格式有软件缺陷，不论如何，它可能会出现在我没有测试的其它的字体格式

软件缺陷的图片信息

- 软件缺陷相关的信息包括软件缺陷的图片、记录信息和如何再现和分离软件缺陷，使开发人员和其他的测试人员更容易分离和重现它。
- 一些涉及用户界面(User Interface)的软件缺陷可能很难用文字清楚地描述，因此软件测试人员通过附上图片比较直观地表示缺陷发生在产品界面什么位置、有什么问题等。 --移动应用测试

软件缺陷报告

- 软件测试进度
- 软件缺陷的描述
  - 软件缺陷的生命周期
  - 严重性和优先级
  - 缺陷的其它属性
  - 完整的缺陷信息
  - 缺陷描述的基本要求
  - 缺陷报告的示例
- 软件缺陷跟踪和分析

软件缺陷的处理和跟踪

- 确保每个被发现的缺陷都能够被解决，“解决”的意思不一定是被修正，也可能是其他处理方式（例如，延迟到下一个版本中修正或者由于技术原因不能被修正）
- 收集缺陷数据并根据缺陷趋势曲线识别测试处于测试过程中的哪个阶段；
- 决定测试过程是否结束，通过缺陷趋势曲线来确定测试过程是否结束是常用并且较为有效的一种方式。

软件缺陷处理技巧

- 审阅。可以由测试管理员、项目管理员或其他人来进行，审阅缺陷报告的质量水平；
- 拒绝。如果审阅者决定需要对一份缺陷报告进行重大修改，应该和测试人员一起讨论，由测试人员纠正缺陷报告，然后再次提交；
- 完善。完整地描述了问题的特征并将其分离，那么审查者就会肯定这个报告；
- 分配。分配给适当的开发人员，如果不知道具体开发人员，应分配给项目开发组长，由开发组长再分配给对应的开发人员；

软件缺陷处理技巧 (2)

- 验证。缺陷的修复需要得到测试人员的验证，同时还要进行回归测试，检查这个缺陷的修复是否会引入新的问题；
- 重新打开。重新打开一个缺陷，需要加注释说明、电话沟通等，否则会引起“打开-修复”多个来回，造成测试人员和开发人员不必要的矛盾
- 关闭。只有测试人员有关闭缺陷的权限，开发人员没有这个权限。
- 暂缓。如果每个人都同意将确实存在的缺陷移到以后处理，应该指定下一个版本号或修改的日期。一旦新的版本开始时，这些暂缓的缺陷应该重新被打开。

缺陷报告

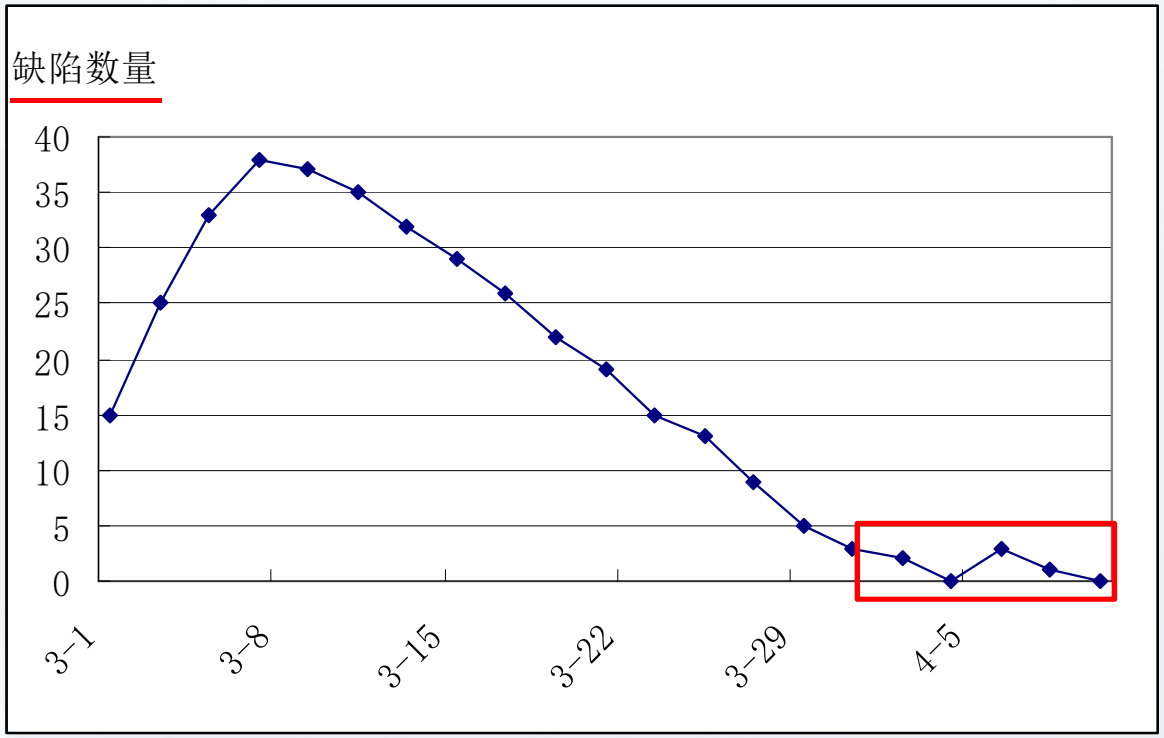
- 缺陷趋势报告，按各种状态将缺陷计数作为时间的函数显示。趋势报告可以是累计的，也可以是非累计的。
- 缺陷分布报告，允许将缺陷计数作为一个或多个缺陷参数的函数来显示，生成缺陷数量与缺陷属性的函数。如缺陷数量和缺陷状态、严重性的分布情况等。
- 缺陷年龄报告，显示缺陷处于活动状态的时间，展示一个缺陷处于某种状态的时间长短，从而了解处理这些缺陷的进度情况。

缺陷趋势分析

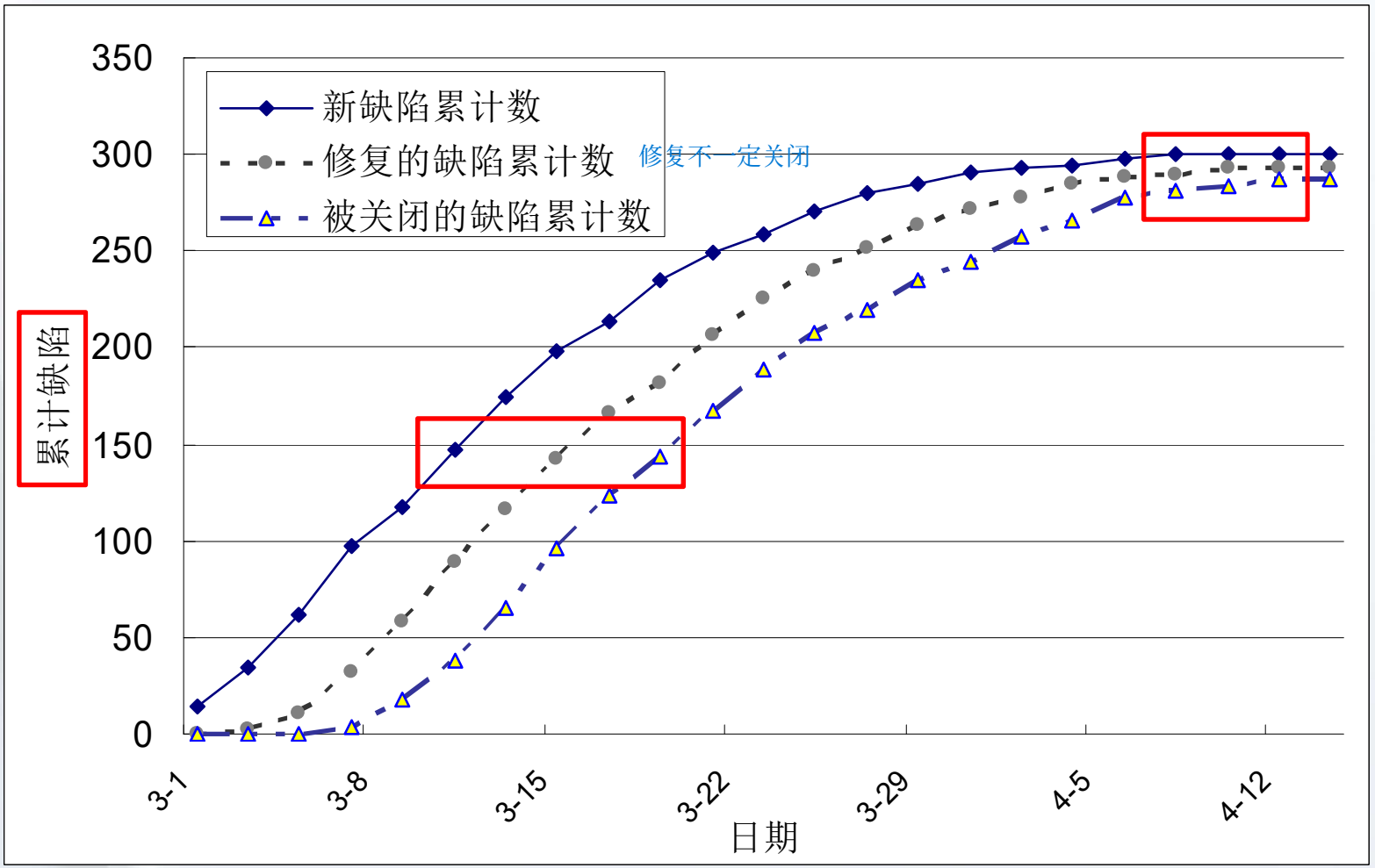
监控（**打开/关闭/已修正的**）缺陷随时间的变化

- 产品开发质量情况取决于累积打开/关闭曲线的趋势
- 项目进度取决于累积打开/关闭曲线起点的时间差
- 开发人员、测试人员的工作进度、效率也能得到反映

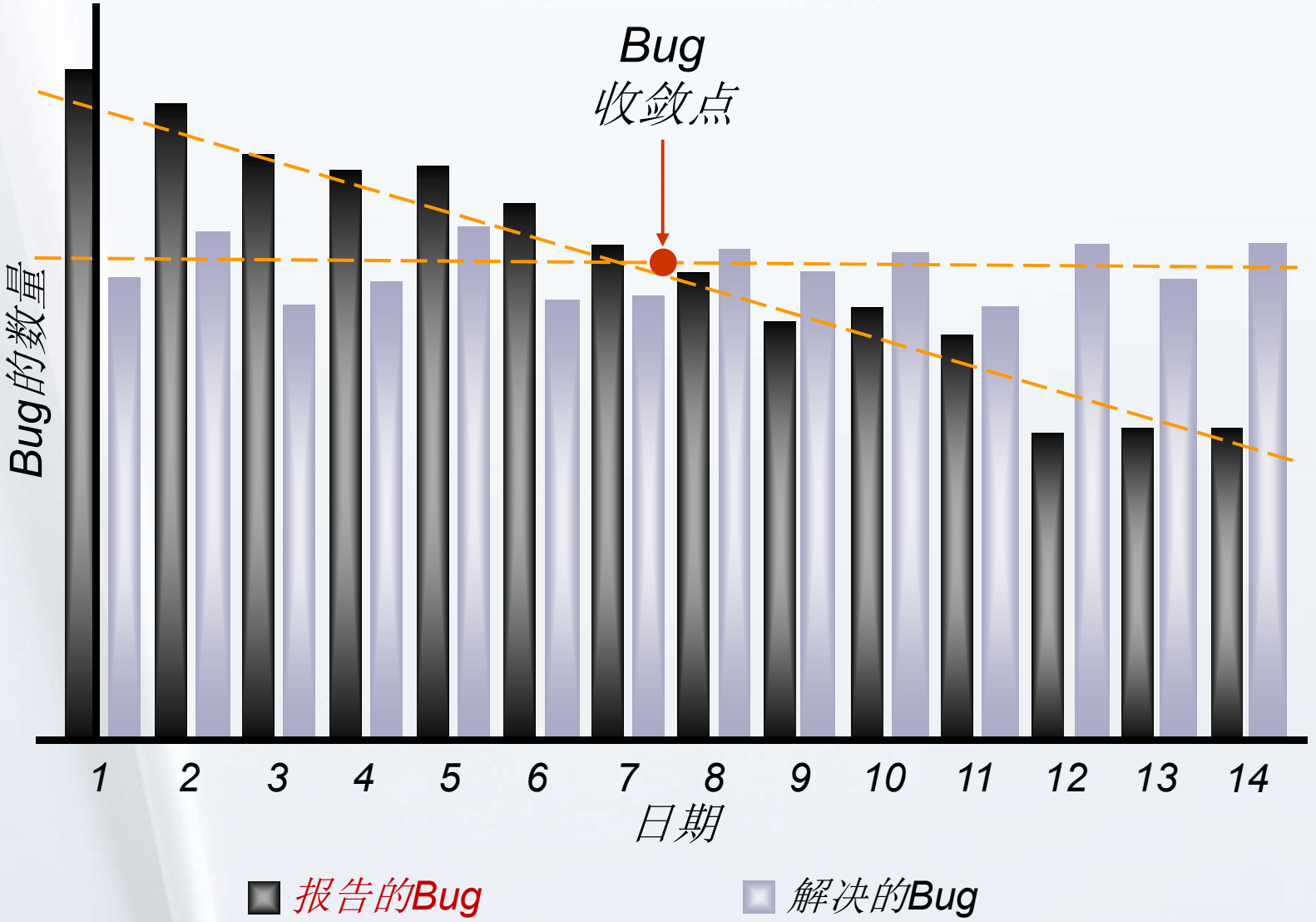
非累积



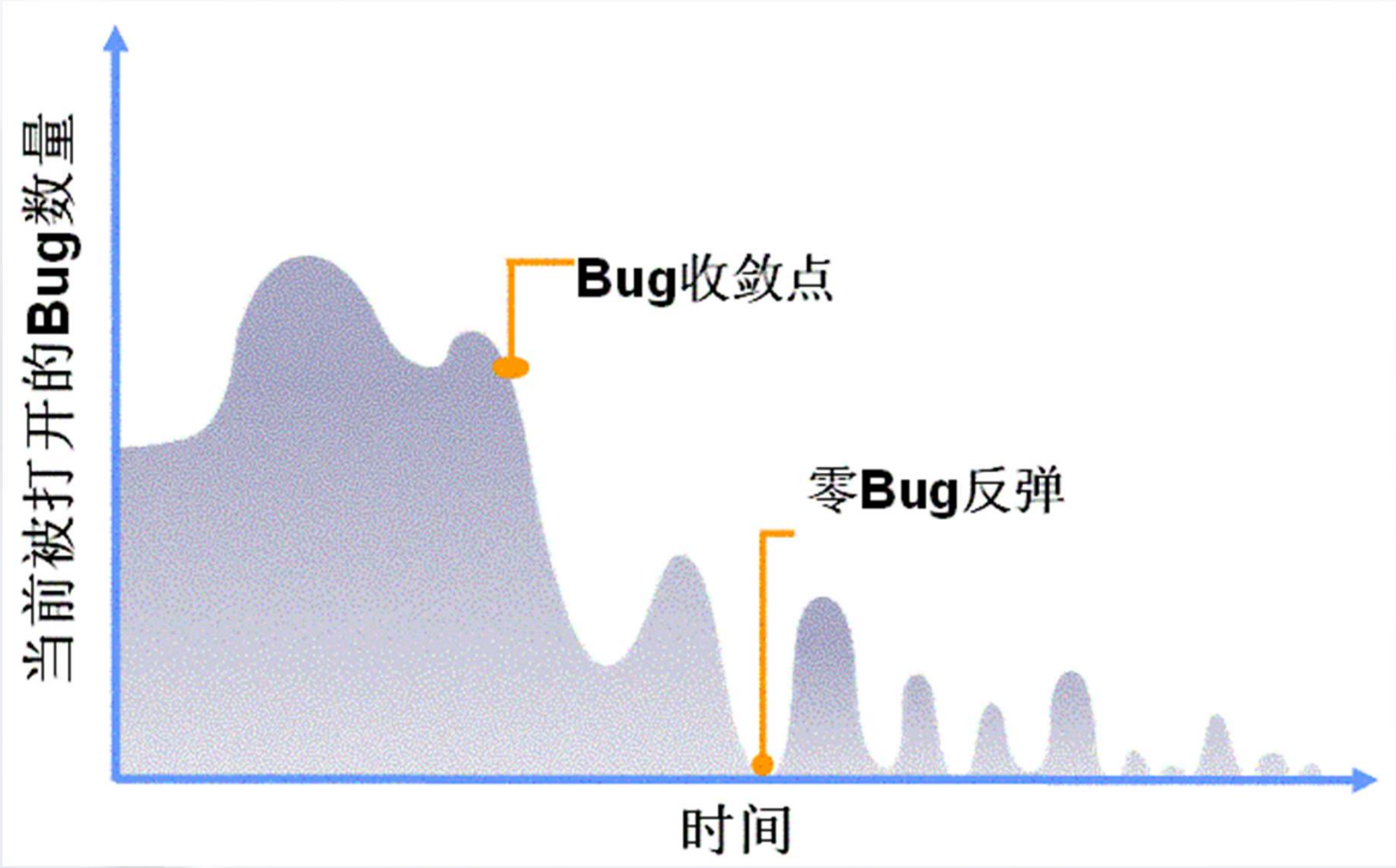
理想趋势图


350
300
250
200
150
100
50
0
累计缺陷
新缺陷累计数
修复的缺陷累计数
被关闭的缺陷累计数
修复不一定关闭
3-1
3-8
3-15
3-22
3-29
4-5
4-12
日期

微软——Bug Converge

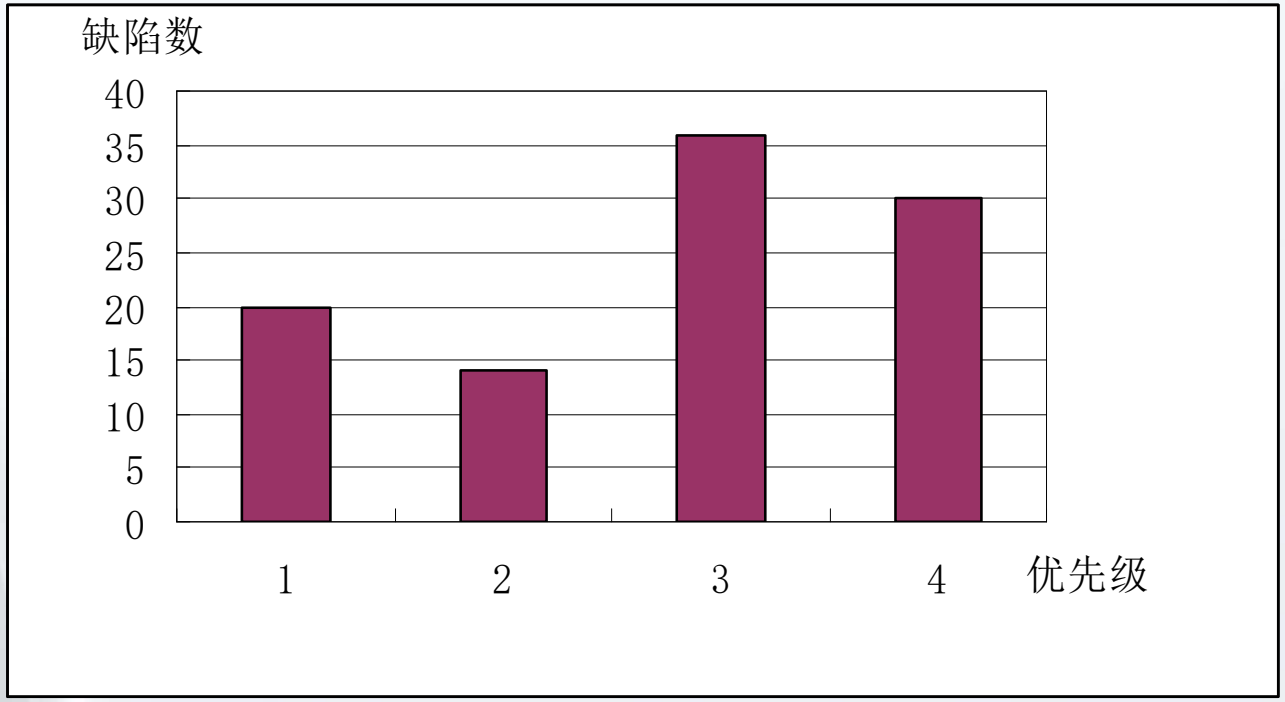


微软公司基于缺陷趋势图的里程碑定义

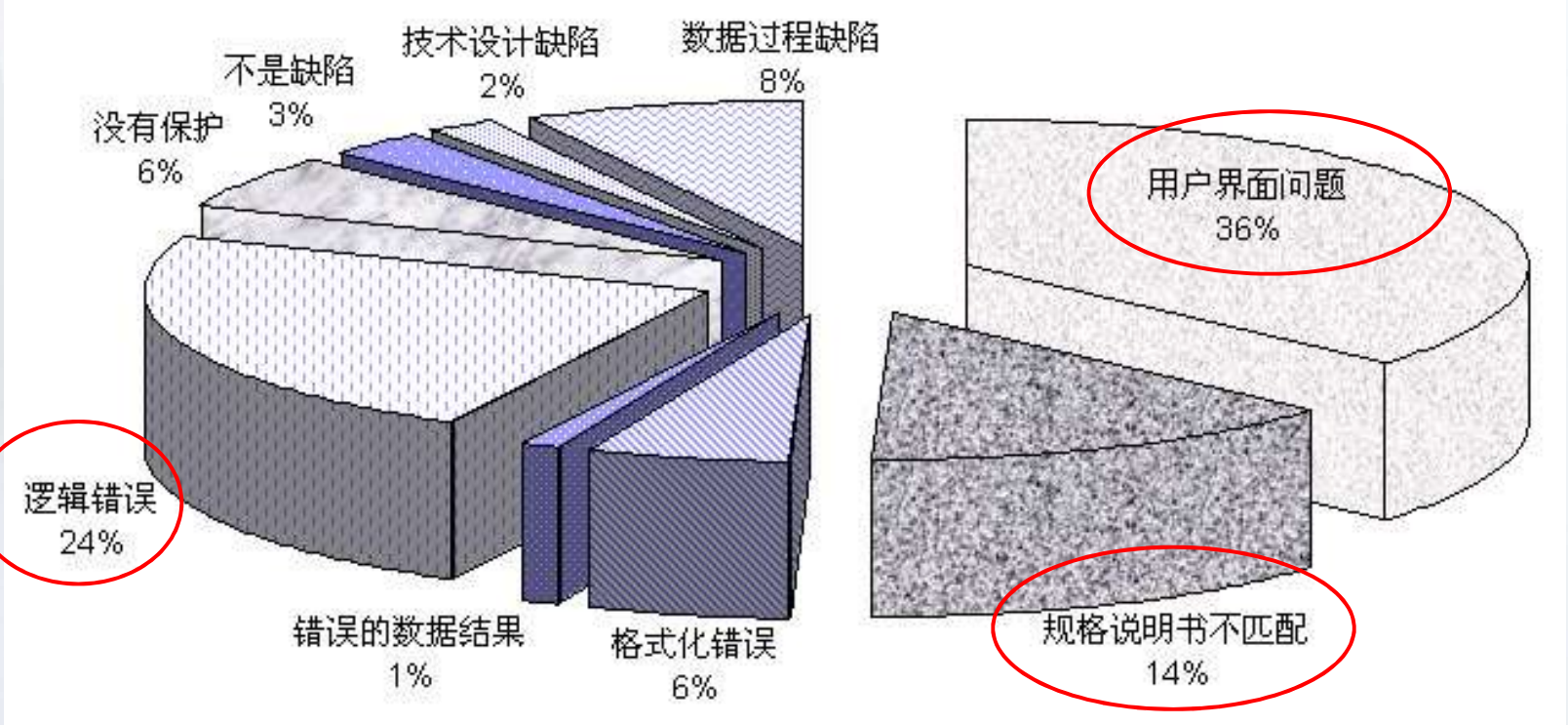


缺陷分布分析

缺陷分布报告，缺陷数量与缺陷属性的函数。如缺陷数量和缺陷状态、严重性的分布情况等。



示例：根本原因图表


技术设计缺陷
2%
数据过程缺陷
8%
不是缺陷
3%
没有保护
6%
用户界面问题
36%
逻辑错误
24%
错误的数据结果
1%
格式化错误
6%
规格说明书不匹配
14%

缺陷跟踪方法

当前缺陷状态 – Bug Dashboard

级别	总数	未处理的	正在处理的	修正的	不是缺陷	重复的	暂不处理	关闭
致命的	2	0	0	0	0	0	0	2
严重的	216	18	7	5	1	4	20	161
一般的	31	23	1	0	0	0	0	7
微小的	5	2	0	0	0	3	0	0

项目发展趋势: 每天的变化、差异，重点进行趋势分析

“三国会议”

- 参加者：项目经理和开发组长、测试组长
- 通过Bug数据库评估每个未解决的Bug
  - ✓ 决定Bug优先级
  - ✓ 可否等到下个里程碑或版本解决？
  - ✓ 谁来解决 Bug triaging
- 预测项目实际进度和发布时间

Bug的优先排列

- 可重复性 (Repeatability)
- 可发生性 (Visibility)
- 严重性 (Severity)
- 优先级=（可重复性+可发生性）X 严重性

必须……

- 尽早定义和推广Bug管理流程
- 明确定义优先级和严重级衡量标准
- 清晰设置Bug提交必须的信息
- 周期性的会诊Bug
- 周期性的检查Bug列表

No ……

- 把全球版本的Bug和本地版本的Bug混为一谈
- 试图隐藏你的Bug
- 解决和关闭你自己的Bug
- 未经诊断发布一个补丁

软件缺陷跟踪系统

- 推动团队内部的有效沟通
- 提供报表和分析
- 按照优先级排列重要Bug
- 跟踪任何一个Bug的整体生命周期
- 纪录任何跟Bug有关的操作
- 报告任何一个Bug的当前状态
- …

开源缺陷跟踪系统

- Mantis，http://mantisbt.sourceforge.net/
- Bugzilla：http://www.mozilla.org/projects/bugzilla/
- Bugzero：http://bugzero.findmysoft.com/
- Scarab：http://scarab.tigris.org/
- TrackIT：http://trackit.sourceforge.net/
- Itracker：http://www.itracker.org/

商业化缺陷跟踪系统

- JIRA：http://www.atlassian.com
- IBM ClearQuest：

  http://www-01.ibm.com/software/awdtools/clearquest/
- Compuware TrackRecord：

  http://www.compuware.com/trackrecord.htm
- HP TestDirector：http://www.hp.com/
- TestTrack Pro：http://www.seapine.com/ttpro.html
- DevTrack：www.techexcel.com/products/devsuite/devtrack.html
- Borland Segue SilkCentral™ Issue Manager等

软件缺陷报告

- 软件测试进度
- 软件缺陷的描述
  - 软件缺陷的生命周期
  - 严重性和优先级
  - 缺陷的其它属性
  - 完整的缺陷信息
  - 缺陷描述的基本要求
  - 缺陷报告的示例
- 软件缺陷跟踪和分析